ISSN 0146-4116, Automatic Control and Computer Sciences, 2014, Vol. 48, No. 6, pp. 313–323. © Allerton Press, Inc., 2014.
Original Russian Text © A.O. Oliinyk, S.Yu. Skrupsky, S.A. Subbotin, 2014, published in Avtomatika i Vychislitel'naya Tekhnika, 2014, No. 6, pp. 5–19.

Using Parallel Random Search to Train Fuzzy Neural Networks

A. O. Oliinyk, S. Yu. Skrupsky, and S. A. Subbotin
Zaporizhzhya National Technical University, Zhukovsky str. 64, Zaporizhzhya, 69063 Ukraine
e-mail: subbotin@zntu.edu.ua

Received April 25, 2014; in final form September 22, 2014

Abstract—A solution to the problem of training fuzzy neural networks is considered. A method of parametric identification of fuzzy neural models that is based on a probabilistic approach when searching for the values of adjustable parameters is proposed. The method allocates the most resource-intensive stages among nodes of a parallel computing system, which reduces the time it takes to adjust the parameters. It is proposed to take into account information on the training sample when forming the initial set of solutions and significance of terms of features, which brings the initial points closer to optimal and accelerates the optimization process.

Keywords: fuzzy neural network, parallel computing, feature, random search, term, evolutionary method, entropy

DOI: 10.3103/S0146411614060078

1. INTRODUCTION

Developing automated systems for the nondestructive quality control of products, diagnostics in engineering and medicine, and pattern recognition has led to the synthesis of models of objects or processes in hand [1–4]. One can efficiently use fuzzy neural networks [5–7], where the fuzzy output system is implemented as a neural network that is easy to analyze, use, and acquire knowledge from, as recognizing models.

To train fuzzy neural networks, one generally uses the method of back propagation of error [5–7] that depends on the choice of the initial search point and requires setting the derivative of the objective function (software implementation makes it difficult to take into account possibility of adjusting parameters of fuzzy neural networks using the objective functions set by the user) and has low speed so that, given limited time or number of iterations, one cannot always achieve acceptable accuracy in practice.

In this work, to train fuzzy neural networks, we propose using random search methods (genetic, evolutionary, and stochastic search methods) [8–11] based on adaptation procedures for some set of solutions (points in the search space) that both allow one to find acceptable solutions without the need to calculate derivatives of the objective function and possess natural computation parallelism.

However, such methods also depend, though much less than gradient methods, on forming the initial set of solutions (choosing initial search points), which is generally done at random, and have low speed of search for optimal solutions. Therefore, it is topical to develop a new method of random search for training (parametric synthesis) of fuzzy neural networks free of these drawbacks. Natural parallelism of computation characteristic of random search methods and their low convergence speed mean it is reasonable to parallelize such methods, given the peculiarities of the problem involved.

The objective of this work is to create a method for training fuzzy neural networks based on parallel random search.

2. STATEMENT OF THE PROBLEM OF TRAINING FUZZY NEURAL NETWORKS

Suppose the set S (1) of observations (precedents) that describe the state of objects or processes involved is given as follows:

$$S = \langle P, T \rangle, \tag{1}$$

where P is the set of features (a feature is a characteristic that describes the object or process involved) and T is the set of values of response (the output parameter).

The sets of values *P* and *T* are represented as matrix (2) and vector (3), respectively, as follows:

$$P = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1m} & \cdots & p_{1M} \\ p_{21} & p_{22} & \cdots & p_{2m} & \cdots & p_{2M} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ p_{q1} & p_{q2} & \cdots & p_{qm} & \cdots & p_{qM} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ p_{Q1} & p_{Q2} & \cdots & p_{Qm} & \cdots & p_{QM} \end{pmatrix}, \qquad (2)$$

$$T = (t_1\ t_2\ \ldots\ t_q\ \ldots\ t_Q)^{\mathrm{tr}}, \qquad (3)$$

where p_{qm} is the value of the *m*th feature of the *q*th observation ($m = 1,2,\ldots,M$, $q = 1,2,\ldots,Q$); t_q is the value of response of the *q*th observation, *M* is the number of features, *Q* is the number of observations, and tr is the symbol of transposition operation.

The structure of a fuzzy neural network is formed by mapping instances of sample (1) into fuzzy production rules. Then, the problem of training the recognizing model based on the fuzzy neural network *NFN* is to identify its parameters (the set of values of weight coefficients *W* and the set of parameters of membership functions μ) so that to ensure the acceptable value of the given quality criterion *G* of the neural model *NFN*. When training fuzzy neural models, one can use, e.g., the recognition error (in problems with the discrete output T) or the mean-square error (when the output parameter *T* can take real values from some range $T \in [t_{\min};t_{\max}]$) as the objective criterion *G*.

3. METHOD OF PARAMETRIC IDENTIFICATION OF FUZZY NEURAL MODELS BASED ON PARALLEL RANDOM SEARCH

As noted above, to train hybrid models of computational intelligence, one can efficiently apply methods of random search with adaptation (methods of genetic, evolutionary, and stochastic search) [8–11].

However, to obtain acceptable results (models that ensure acceptable recognition or prediction accuracy), it is not generally sufficient to run such methods once due to the probabilistic approach used and, in some cases, cycling in local optima.

The need for multiple runs and significant search time make it reasonable to parallelize the training process for hybrid intelligent models, including fuzzy neural networks.

To develop a parallel random search method for training fuzzy neural models, we single out stages of random optimization that it is reasonable to parallelize. It is known from the literature [8–11] that the principal stages of probabilistic methods are as follows:

• initializing the initial set of solutions $R^{(0)} = \{\chi_1^{(0)},\chi_2^{(0)},\ldots,\chi_{N\chi}^{(0)}\}$, where

$\chi_k^{(0)} = \{g_{1k}^{(0)},g_{2k}^{(0)},\ldots,g_{N_g k}^{(0)}\}$ is the *k*th solution in the set $R^{(0)}$, $k = 1,2,\ldots,N_\chi$;

N_χ is the number of elements of the set $R^{(0)}$ (the number of randomly generated solutions in the course of initialization), the variable N_χ does not generally change in the course of probabilistic optimization;

$g_{lk}^{(0)}$ is the value of the *l*th element (parameter) in the *k*th solution, $l = 1,2,\ldots,N_g$;

N_g is the number of parameters in the solution $\chi_k^{(i)}$;

• estimating the current set of solutions $G^{(i)} = G(R^{(i)})$ and checking the stopping criteria, where *G* is the objective function that helps estimate the quality of the *i*th set of solutions $R^{(i)}$;

• forming a new set of solutions $R^{(i+1)}$, which, for an evolutionary search, is performed by applying the crossover and mutation operators.

To reduce the time of the random search when constructing hybrid models of computational intelligence, it is reasonable to use a priori information on the training sample at the initialization stage when generating the initial set of solutions. Hence, this approach requires one to synthesize fuzzy neural models N_χ times (forming the set of initial values of their parameters), which results in the significant use of computational resources and implements the parallelization of this stage.

Generally, the stage of estimating the current set of solutions $R^{(i)}$ is the most resource intensive, as it uses many computational and time resources to calculate values of the objective function *G* for each *k*th

($k = 1,2,\ldots,N_\chi$) solution $\chi_k^{(i)}$ in the form $G_k^{(i)} = G(\chi_k^{(i)})$. Since this stage is of high computational complexity, is performed slowly and does not require data exchange between solutions $\chi_k^{(i)}$, it is reasonable to be performed in parallel.

Generating a new set of solutions $R^{(i+1)}$ is an important stage. To study the domains of local optima in more detail, it is reasonable to decompose the current set of solutions $R^{(i)}$ into subsets $R^{(i,j)}$ and then search for the optimum in each of them: $R^{(i)} \rightarrow \{R^{(i,1)}, R^{(i,2)},\ldots, R^{(i,N_{\text{pr}})}\}$, where $R^{(i,j)}$ is the jth subset of the ith set $R^{(i)}$ and N_{pr} is the number of processes that are simultaneously run in the parallel computing system. We propose to perform this decomposition using the a priori information on location of the solutions $\chi_k^{(i)}$ in the space of elements g_l ($l = 1,2,\ldots,N_g$). Unlike the island model of evolutionary search [8, 9], which implies the random generation of subpopulations $R^{(i,j)}$, with this approach, one can take into account information on spatial location of solutions $\chi_k^{(i)}$ in the set $R^{(i)}$ and study domains of possible optima in more detail. We propose to perform random search with adaptation in each jth subset $R^{(i,j)}$ on the jth process of the parallel computing system ($j = 1,2,\ldots,N_{\text{pr}}$) for N_{it} iterations.

After N_{it} iterations of a random search over each subset $R^{(i,j)}$, they are combined into one population $R^{(i')} = R^{(i',1)} \cup R^{(i',2)} \cup \ldots \cup R^{(i',N_{\text{pr}})}$, followed by the probabilistic optimization over the combined set. This will allow one to detect new domains that have local (and, may be, global) optima in them. To reduce the time of probabilistic optimization when operating with a combined set of solutions, we propose calculating the values of the objective function $G_k^{(i)}$ of the solutions $\chi_k^{(i)}$ based on the processes in the parallel system (i.e., by parallelizing it).

Therefore, the principal stages of the proposed method are the initialization, the estimation of solutions, the decomposition of the current set of solutions into subsets, and the search for optimal solutions based on a parallel approach. Moreover, to be able to use a random search to train fuzzy neural models, we need to find a way to represent synthesized neural models like the solutions χ_k.

Thus, the proposed method of training fuzzy neural networks is based on the probabilistic approach when searching for values of adjustable parameters and distributes most resource intensive stages between nodes of the parallel computing system, which reduces the time to adjust the parameters of the neural models to be synthesized, including the weight coefficients and parameters of membership functions of neuroelements.

3.1. Encoding Solutions

Set of observations (1) and the model structure are the input information used to train hybrid models of computational intelligence. The number of adjustable parameters N_g (parameters of the membership functions and weight coefficients of the neuroelements used) of the fuzzy neural models depends on the number of neurons used, which in turn depends on the production rules that describe the set S.

To use a random search for the parametric identification of fuzzy neural models, we present a way to represent the values of adjustable parameters g_{lk} in the solutions (chromosomes for the evolutionary search) χ_k. Suppose N_R production rules are singled out from the sample $S = \langle P,T \rangle$. It is reasonable to use the ANFIS fuzzy neural network [5, 12], which is now widely used in solving pattern recognition problems [4, 5, 12, 13], and implements the Takagi–Sugeno fuzzy logic inference system, as a feedforward neural network that consists of five layers [5, 12] as the basis for the model to be synthesized.

The ANFIS-based synthesis of the model implies the selection of values of parameters of fuzzy terms and rules. The developed method can be similarly applied for the parametric synthesis of other types of hybrid models of computational intelligence (to do this, it will be sufficient to determine the adjustable parameters and the way to represent them in the solution χ_k).

Outputs of the first-layer neurons of the ANFIS fuzzy neural network show whether the instance to be recognized belongs to fuzzy terms of features with the membership functions $\mu_{mn}^{(1)}$ ($\mu_{mn}^{(1)}$ is the membership function of the instance to be recognized to the nth term of the mth feature p_m, $m = 1,2,\ldots,M$, $n = 1,2,\ldots,N_{\text{int}}$, N_{int} is the number of terms of features). We define the term as an interval of values of the feature that corresponds to some concept, category, or projection of the cluster to the feature's axis.

As the membership functions of the first-layer neurons, it is reasonable to choose inverted U-shaped (4) or trapezoidal (5) [9, 14, 15] functions, since they are universal with regard to the class of elementary membership functions, i.e., they can be used to construct other functions, as follows:

$$\mu_{mn}^{(1)} = \mu_{S\,mn}^{(1)}\mu_{Z\,mn}^{(1)}, \tag{4}$$

$$\mu_{mn}^{(1)} = \begin{cases} 0, & p_m < c_{mn}; \\ (p_m - c_{mn})(a_{mn} - c_{mn})^{-1}, & c_{mn} \le p_m < a_{mn}; \\ 1, & a_{mn} \le p_m \le b_{mn}; \\ (d_{mn} - p_m)(d_{mn} - b_{mn})^{-1}, & b_{mn} < p_m \le d_{mn}; \\ 0, & p_m > d_{mn}, \end{cases} \tag{5}$$

where a_{mn}, b_{mn}, c_{mn}, and d_{mn} are the parameters of the membership function μ_{mn}. Furthermore, $\mu_{S\,mn}^{(1)}$ and $\mu_{Z\,mn}^{(1)}$ are the S- (6) and Z-shaped (7) membership functions

$$\mu_{S\,mn}^{(1)} = \begin{cases} 0, & p_m < c_{mn}; \\ (p_m - c_{mn})(a_{mn} - c_{mn})^{-1}, & c_{mn} \le p_m < a_{mn}; \\ 1, & p_m \ge a_{mn}, \end{cases} \tag{6}$$

$$\mu_{Z\,mn}^{(1)} = \begin{cases} 1, & p_m \le b_{mn}; \\ (d_{mn} - p_m)(d_{mn} - b_{mn})^{-1}, & b_{mn} < p_m \le d_{mn}; \\ 0, & p_m > d_{mn}. \end{cases} \tag{7}$$

The parameters a_{mn} and b_{mn} of the functions $\mu_{mn}^{(1)}$ are chosen based on the boundaries of decomposition intervals of the values of the feature $p_m \in [p_{m\min}; p_{m\max}]$. Therefore, for each of MN_{int} first-layer neurons, c_{mn} and d_{mn} will be the adjustable parameters.

The second layer implements left-hand sides (antecedents) of the rules. One can use formula (8) to calculate the outputs of the neurons of this layer as follows:

$$\mu_r^{(2)} = \prod_{m=1}^{M}\prod_{n=1}^{N_{\text{int}}} w_{mn}^{(2,r)}\mu_{mn}^{(1)}, \quad r = 1,2,\ldots,N_R, \tag{8}$$

where $w_{mn}^{(2,r)}$ are the parameters that show there is a link between the respective first- and second-layer neurons and are found from the given structure of the network ($w_{mn}^{(2,r)} = 0$ if the nth term of the mth feature is not in the antecedent of the rth rule; $w_{mn}^{(2,r)} = 1$ if the respective term of the feature is in the hypothesis of the rth rule). In addition to formula (8), one can find the values of outputs of the second-layer neurons $w_{mn}^{(2,r)}$ as the minimum of products $w_{mn}^{(2,r)}\mu_{mn}^{(1)}$, $m = 1,2,\ldots,M$, $n = 1,2,\ldots,N_{\text{int}}$.

The third-layer neurons normalize the values of the degrees of fulfillment of antecedents of rules (9)

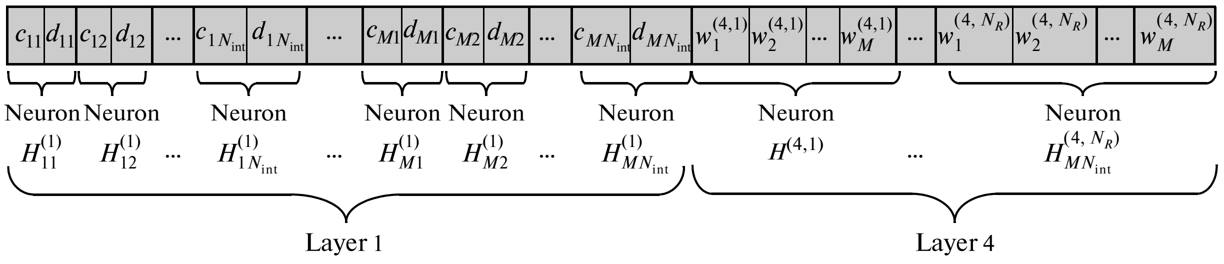
$$\mu_r^{(3)} = \mu_r^{(2)}\left(\sum_{z=1}^{N_R}\mu_z^{(2)}\right)^{-1}, \quad r = 1,2,\ldots,N_R. \tag{9}$$

Hence, the second and third layers have no parameters to be adjusted with random search.

Every rth neuron of the fourth layer can be calculated as follows:

$$y_r = \mu_r^{(4)} = \mu_r^{(3)}\sum_{m=1}^{M} w_m^{(4,r)} p_m, \quad r = 1,2,\ldots,N_R, \tag{10}$$

where $w_m^{(4,r)}$ are the weight coefficients that give significance of the feature p_m in the rth rule and are adjustable parameters of the fuzzy neural model.


c_{11} d_{11} c_{12} d_{12} ... $c_{1N_{\text{int}}}$ $d_{1N_{\text{int}}}$ ... c_{M1} d_{M1} c_{M2} d_{M2} ... $c_{MN_{\text{int}}}$ $d_{MN_{\text{int}}}$ $w_1^{(4,1)}$ $w_2^{(4,1)}$ ... $w_M^{(4,1)}$ ... $w_1^{(4,N_R)}$ $w_2^{(4,N_R)}$ ... $w_M^{(4,N_R)}$
Neuron $H_{11}^{(1)}$
Neuron $H_{12}^{(1)}$
Neuron $H_{1N_{\text{int}}}^{(1)}$
Neuron $H_{M1}^{(1)}$
Neuron $H_{M2}^{(1)}$
Neuron $H_{MN_{\text{int}}}^{(1)}$
Neuron $H^{(4,1)}$
Neuron $H_{MN_{\text{int}}}^{(4,N_R)}$
Layer 1
Layer 4

Schematic representation of the solution χ_k for parametric identification of fuzzy neural networks.

The fifth layer has one neuron that calculates the total output of the network (11)

$$y = \sum_{r=1}^{N_R} y_r. \tag{11}$$

Thus, when training fuzzy neural models, the adjustable parameters are MN_{int}, parameters c_{mn} and d_{mn} of the membership functions $\mu_{mn}^{(1)}$ of the first-layer neurons, and MN_R weight coefficients $w_m^{(4,r)}$ of the fourth-layer neurons. Hence, the total number of adjustable parameters N_g can be found by formula (12)

$$N_g = MN_{\text{int}} + MN_R = M(N_{\text{int}} + N_R). \tag{12}$$

To train fuzzy neural networks, given that it is reasonable to only adjust parameters of the first and fourth layers, we can represent the solution (chromosome) χ_k as shown in figure.

The proposed means of representation (figure) allows one to carry out the parametric identification of fuzzy neural networks using a random search.

3.2. Initialization

At the initialization stage of the proposed method, we give the input parameters (the set $S = \langle P,T \rangle$, the structure of the fuzzy neural network, which requires parametric identification, and a number of parameters needed for random search). To bring the initial search points $\chi_k^{(0)} = \{g_{1k}^{(0)}, g_{2k}^{(0)}, \dots, g_{N_g k}^{(0)}\}(k = 1,2,\dots,N_\chi)$ closer to optimal, we form the set $R^{(0)}$, unlike the known random search methods [8–11, 13], given the a priori information on the training sample $S = \langle P,T \rangle$. To do this, we decompose the range of values $\Delta p_m = [p_{m\min}; p_{m\max}]$ of each mth feature p_m $(m = 1,2,\dots,M)$ into the given number of intervals N_{int}. Thus, we form the intervals $\Delta p_{mn} = [p_{mn\min}; p_{mn\max}]$ $(n = 1,2,\dots,N_{\text{int}})$, the boundaries of which help determine the parameters of fuzzy terms.

When calculating the values of parameters c_{mn} and d_{mn}, we take into account the significance of the nth term Δp_{mn} of the mth feature p_m in order to distinguish the instances of the sample $S = \langle P,T \rangle$. We define the significance (informativity, importance) V_{mn} of the nth term of the mth feature as the product of the variables $\frac{Q|_{p_m \in \Delta p_{mn}}}{Q}$ and $(1 - \text{Entr}(\Delta p_{mn}))$, which characterize the location density of the instances of the set S in the interval Δp_{mn} of the feature p_m and the degree of influence of the term Δp_{mn}, respectively, based on the value of the output parameter T (13) as follows:

$$V_{mn} = \frac{Q|_{p_m \in \Delta p_{mn}}}{Q}(1 - \text{Entr}(\Delta p_{mn})), \tag{13}$$

where $Q|_{p_m \in \Delta p_{mn}}$ is the number of observations of the sample S, the values of the mth features p_m of which are in the interval Δp_{mn}, and $\text{Entr}(\Delta p_{mn})$ is the entropy of the interval Δp_{mn}, i.e., the variable [2, 9, 10] that characterizes the uncertainty measure of the output parameter T given that $p_m \in \Delta p_{mn}$. The higher its entropy, the less informative the interval is for making decisions. The variable $\mathrm{Entr}(\Delta p_{mn})$ can be calculated by as follows:

$$\mathrm{Entr}(\Delta p_{mn}) = -\sum_{q=1}^{N_{Q,p_m \in \Delta p_{mn}}} \rho_q \log \rho_q, \tag{14}$$

where $N_{Q,p_m \in \Delta p_{mn}}$ is the number of decomposition intervals (classes) of the output parameter T of the fragment of the sample $S_{p_m \in \Delta p_{mn}}$, in which the values of the mth features p_m are in the interval Δp_{mn}, and ρ_q is the probability (15) that the output parameter T takes the value q in the set $S_{p_m \in \Delta p_{mn}}$ (given that $p_m \in \Delta p_{mn}$)

$$\rho_q = \frac{Q|_{T=q;S_{p_m \in \Delta p_{mn}}}}{Q|_{p_m \in \Delta p_{mn}}}. \tag{15}$$

Here, $Q|_{T=q;S_{p_m \in \Delta p_{mn}}}$ is the number of observations in the set $S_{p_m \in \Delta p_{mn}}$ in which the value of the output parameter T is q $(T = q)$.

Along with the other required parameters, the estimates V_{mn} calculated on the main process are sent to other processes for the parallel generation of the initial set of solutions $R^{(0)} = \{\chi_1^{(0)}, \chi_2^{(0)}, \ldots, \chi_{N\chi}^{(0)}\}$.

When constructing the solutions $\chi_k^{(0)}$, we expand the boundaries of the intervals $[c_{mn}; a_{mn}]$ and $[b_{mn}; d_{mn}]$ by decreasing and increasing the values of parameters c_{mn} and d_{mn}, respectively, depending on the variables V_{mn} that characterize the significance of the nth term of the mth feature p_m. We expand the intervals that correspond to high values of the estimates V_{mn}. To form a set of solutions in the form of N_χ sets of parameters of fuzzy neural models $R^{(0)} = \{\chi_1^{(0)}, \chi_2^{(0)}, \ldots, \chi_{N\chi}^{(0)}\}$ rather than a single solution, to ensure variability, we introduce the coefficient rand[x; y] (a randomly generated number in the interval from x to y), which ensures the stochastic component when calculating the parameters c_{mn} and d_{mn} of the fuzzy neural models to be synthesized. Given the above, we calculate the values of parameters c_{mn} and d_{mn} as follows:

$$c_{mn} = a_{mn} - \frac{p_{m\max} - p_{m\min}}{N_{\mathrm{int}}} V_{mn} \mathrm{rand}[1;2], \tag{16}$$

$$d_{mn} = b_{mn} + \frac{p_{m\max} - p_{m\min}}{N_{\mathrm{int}}} V_{mn} \mathrm{rand}[1;2]. \tag{17}$$

Using these formulas to calculate the values of c_{mn} and d_{mn}, one can generate the set of respective parameters, each of which depends on the estimate of the significance of its term Δp_{mn} and a random number in the interval from 1 to 2, which ensures the possibility of a maximum increase in the lengths of the intervals $[c_{mn}; a_{mn}]$ and $[b_{mn}; d_{mn}]$ up to the width of two intervals Δp_{mn}. This makes it possible for the neighboring intervals Δp_{mn-2}, Δp_{mn-1}, Δp_{mn}, Δp_{mn+1}, Δp_{mn+2} of the membership functions $\mu_{mn}^{(1)}$ to cross, which in turn allows one to introduce fuzziness at the first layer of the fuzzy neural network to be synthesized. If we need to ensure more intervals Δp_{mn} cross and to introduce greater fuzziness, we can increase the value of the second parameter in the function rand[x; y] in formulas (16) and (17) given above.

To calculate the weight coefficients $w_m^{(4,r)}$ of the fourth layer of the fuzzy neural network to be synthesized, which define significance of the feature p_m in the rth rule, we use the estimates V_{mn} obtained earlier. The significance of the feature p_m in the rth rule $V_m(\mathrm{rule}_r)$ is given by informativity $V_{\substack{mn,\\ \Delta p_{mn} \in \mathrm{rule}_r}}$ of the respective term Δp_{mn} of the mth feature in this rule. To ensure variability in order to form a set of solutions when calculating the values $w_m^{(4,r)}$, we also take into account the stochastic component rand[x; y] as follows:

$$w_m^{(4,r)} = \left(1 + V_{\substack{mn,\\ \Delta p_{mn} \in \mathrm{rule}_r}}\right) \mathrm{rand}[0;1]. \tag{18}$$

Thus, at each jth node $(j = 1, 2, \ldots, N_{\mathrm{pr}})$ of the parallel system, the subsets $R^{(0,j)} = \left\{\chi_1^{(0,j)}, \chi_2^{(0,j)}, \ldots, \chi_{|R^{(0,j)}|}^{(0,j)}\right\}$ of solutions are generated, where the parameters $g_{lk}^{(0,j)}$ of the solution $\chi_k^{(0,j)} = \left\{g_{1k}^{(0,j)}, g_{2k}^{(0,j)}, \ldots, g_{N_g k}^{(0,j)}\right\}$ are calculated by formulas (16)–(18).

3.3. Estimation of Solutions

Then, we perform transformation (19)

$$\chi_k^{(0,j)} \to NFN_k^{(0,j)} \tag{19}$$

that, using expressions (4)–(11), leads to the generation of the fuzzy neural network $NFN_k^{(0,j)}$ based on the given structure and the set of parameters represented in the solution $\chi_k^{(0,j)}$.

Then, we estimate the recognition quality for each *k*th generated fuzzy neural model $NFN_k^{(0,j)}$: $G(NFN_k^{(0,j)})$. This estimate is calculated as the recognition error (if the output T is discrete in the set S) or as the root-mean-square error (if the output T takes real values), which requires using the model $NFN_k^{(0,j)}$ to be synthesized to calculate the values of the output parameter $T_q(NFN_k^{(0,j)})$ for each *q*th observation of the set $S = \langle P,T\rangle$. The resulting estimate $G(NFN_k^{(0,j)})$ characterizes the fitness $G_k^{(0,j)}$ of the solution $\chi_k^{(0,j)}$ in the set $R^{(0)}$

$$G_k^{(0,j)} = G(\chi_k^{(0,j)}) = G(NFN_k^{(0,j)})$$

so that, in the future, we can select the best-fitted solutions to generate new sets $R^{(i+1)}$.

3.4. Decomposition of Current Set of Solutions into Subsets

After the quality estimates $G^{(i)} = G\left(R^{(i)}\right)$ of the current set of solutions $R^{(i)} = \left\{\chi_1^{(i)},\chi_2^{(i)},\dots,\chi_{N\chi}^{(i)}\right\}$ are calculated, the set is decomposed into subsets $R^{(i,j)}$ as follows:

$$R^{(i)} \to \left\{R^{(i,1)},R^{(i,2)},\dots,R^{(i,N_{\rm pr})}\right\}.$$

Then, the extrema of the objective function $G^{(i)} = G(R^{(i)})$ are searched for in each *j*th subset $R^{(i,j)}$ at the *j*th process of the parallel system. We form the subsets $R^{(i,j)}$ so that each *j*th set is a group of densely located solutions $\chi_k^{(i)}$ in the space of elements $\{g_1,g_2,\dots,g_{N_g}\}$. This allows one to take into account the a priori information on the spatial location of the solutions $\chi_k^{(i)}$ and study the domains of the location of extremal solutions in more detail using random searches. Since we propose processing each set $R^{(i,j)}$ on the respective process, the total number of clusters (subsets $R^{(i,j)}$) formed corresponds to the number of processes of the parallel system $N_{\rm pr}$.

To determine groups $R^{(i,j)}$ of single-type solutions $\chi_k^{(i)}$, we use a priori information on the initial location of the adjustable parameters g_{lk} (the parameters c_{mn} and d_{mn} of the membership functions $\mu_{mn}^{(1)}$ of first-layer neuroelements and values of weight coefficients $w_m^{(4,r)}$ of the fourth-layer neurons). Formulas (16), (17) given above show that these parameters generally occur in the intervals

$$c_{mn} \in [a_{mn} - 2\Delta p_{mn}; a_{mn}],\ d_{mn} \in [b_{mn}; b_{mn} + 2\Delta p_{mn}]\ \text{(for } V_{mn} = 1 \text{ and } \mathrm{rand}[1;2] = 2\text{) and } w_m^{(4,r)} \in [0;2].$$

Therefore, when forming the decomposition $R^{(i)} \to \left\{R^{(i,1)},R^{(i,2)},\dots,R^{(i,N_{\rm pr})}\right\}$, we propose taking into account the location of the solutions $\chi_k^{(i)}$ in the respective intervals. We decompose each of these intervals into $N_{\rm pr}$ segments. We use formula (20) to estimate the rank $\mathrm{Rg}(\chi_k^{(i)})$ of the solution $\chi_k^{(i)}$

$$\mathrm{Rg}\left(\chi_k^{(i)}\right) = \sum_{l=1}^{N_g}\mathrm{Rg}\left(g_{lk}^{(i)}\right), \tag{20}$$

where $\mathrm{Rg}(g_{lk}^{(i)})$ is the rank of the solution $\chi_k^{(i)}$ with respect to the *l*th feature and corresponds to the number of the decomposition interval of the range of values of the parameter $g_{lk} \in [g_{lk\,\min}; g_{lk\,\max}]$ (depending on the way the solution $\chi_k^{(i)}$ is represented, the parameter and the range of its values g_{lk} corresponds to one of three values c_{mn}, d_{mn}, $w_m^{(4,r)}$). We propose calculating the variable $\mathrm{Rg}(g_{lk}^{(i)})$ by formula (21) as follows:

$$\mathrm{Rg}\left(g_{lk}^{(i)}\right) = 1 + \mathrm{ceil}\left(\frac{g_{lk\min} - g_{lk\max}}{N_{\mathrm{pr}}}\right), \tag{21}$$

where ceil(A) is the function that returns the integer part of A.

Using formula (21), we can find the ranks $\mathrm{Rg}(g_{lk}^{(i)})$ so that they correspond to the numbers of ranges of decomposition of the interval $[g_{lk\min}; g_{lk\max}]$ into N_{pr} segments and can take values $1, 2, \ldots, N_{\mathrm{pr}}$. No need in computationally complex sorting operations is also why it is reasonable to apply this approach in order to estimate the solutions $\chi_k^{(i)}$ (using the ranks $\mathrm{Rg}(\chi_k^{(i)})$) for them to be decomposed into subsets $R^{(i,j)}$.

The set $\{R^{(i,1)}, R^{(i,2)}, \ldots, R^{(i,N_{\mathrm{pr}})}\}$ is formed as follows. The subset $R^{(i,1)}$ is ascribed with the first N_χ / N_{pr} solutions $\chi_k^{(i)}$ chosen with respect to the values of the variable $\mathrm{Rg}(\chi_k^{(i)})$. The next N_χ / N_{pr} solutions form the subset $R^{(i,2)}$. This process is continued similarly for each subset $R^{(i,j)}$ until the last N_{pr} th subset $R^{(i,N_{\mathrm{pr}})}$ is formed.

3.5. *Search for Optimal Solutions Based on Parallel Approach*

After the set $R^{(i)}$ is decomposed into subsets $\{R^{(i,1)}, R^{(i,2)}, \ldots, R^{(i,N_{\mathrm{pr}})}\}$ on the main process, they are sent to other processes for parallel random search for extrema.

It is known [8–11] that the principal stages of a random search consist of forming a new set of solutions, estimating the quality of new solutions, and checking the stopping criteria.

To form the new set $R^{(i',j)}$ on the jth process, we select the solutions $\chi_k^{(i,j)} \in R^{(i,j)}$ based on the values of their objective function $G_k^{(i,j)} = G(\chi_k^{(i,j)})$. We propose to introduce the solution $\chi_k^{(i,j)} \in R^{(i,j)}$ into the set of solutions $R_{\mathrm{selected}}^{(i,j)}$ accepted for forming the new generation $R^{(i',j)}$ given that the following condition (22):

$$G_{k,\mathrm{norm}}^{(i,j)} \le \mathrm{rand}[0;1], \tag{22}$$

where $G_{k,\mathrm{norm}}^{(i,j)}$ is the normalized value of the objective function $G_k^{(i,j)}$ for the set $R^{(i,j)}$. Thus, the smaller the error $G_k^{(i,j)}$ of the network $NFN_k^{(i,j)} = NFN(\chi_k^{(i,j)})$, the higher the probability that the solution $\chi_k^{(i,j)} \in R^{(i,j)}$ will be selected to form the new set $R^{(i',j)}$.

Similar to the evolutionary approach to searching optimal solutions [8, 10, 11], we propose forming a new set of solutions $R^{(i',j)}$ out of most adapted sets of $\chi_k^{(i,j)}$ (elite ones) and the sets resulted from applying evolutionary crossover and mutation operators. The number of solutions of new generations obtained in a given way is determined by rounding off the integer as follows:

$$N_{\mathrm{elite}} = \alpha N_\chi,\ \ N_{\mathrm{cross}} = \beta N_\chi,\ \ N_{\mathrm{mutation}} = \gamma N_\chi, \tag{23}$$

where $N_{\mathrm{elite}} = \alpha N_\chi$ is the number of elite solutions (the solutions $\chi_k^{(i,j)} \in R^{(i,j)}$ with the best values of the estimates $G_k^{(i,j)}$ in the set $R^{(i,j)}$); $N_{\mathrm{cross}} = \beta N_\chi$ is the number of solutions resulting from the crossover operator; $N_{\mathrm{mutation}} = \gamma N_\chi$ is the number of solutions generated by the mutation operator; and α, β, γ are the coefficients that indicate the significance of a given way of obtaining the new set, where $\alpha + \beta + \gamma = 1$.

When the new set is formed, N_{cross} solutions are generated by crossover of sets represented in the set $R_{\mathrm{selected}}^{(i,j)}$. The elements $g_{lk}^{(i')}$ of the new solution $\chi_k^{(i',j)}$ based on parent solutions $\chi_{p1}^{(i,j)} \in R_{\mathrm{selected}}^{(i,j)}$ and $\chi_{p2}^{(i,j)} \in R_{\mathrm{selected}}^{(i,j)}$ can be determined by formulas (24)–(27) as follows:

$$g_{lk}^{(i')} = \eta g_{l,p1}^{(i)} + (1-\eta) g_{l,p2}^{(i)}, \tag{24}$$

$$g_{lk}^{(i')} = \max\left(g_{l,p1}^{(i)}, g_{l,p2}^{(i)}\right) - \eta\left(\max\left(g_{l,p1}^{(i)}, g_{l,p2}^{(i)}\right) - g_{l\min}\right) = \eta g_{l\min} + (1-\eta)\max\left(g_{l,p1}^{(i)}, g_{l,p2}^{(i)}\right), \tag{25}$$

$$g_{lk}^{(i')} = \min\left(g_{l,p1}^{(i)}, g_{l,p2}^{(i)}\right) + \eta\left(g_{l\max} - \min\left(g_{l,p1}^{(i)}, g_{l,p2}^{(i)}\right)\right) = \eta g_{l\max} + (1-\eta)\min\left(g_{l,p1}^{(i)}, g_{l,p2}^{(i)}\right), \tag{26}$$

$$g_{lk}^{(i')} = \frac{1}{2}\Big((1-\eta)(g_{l\min} + g_{l\max}) + \eta\Big(g_{l,p1}^{(i)} + g_{l,p2}^{(i)}\Big)\Big), \tag{27}$$

where $g_{l,p1}^{(i)}$ and $g_{l,p2}^{(i)}$ are the values of the lth elements of the parent solutions $\chi_{p1}^{(i,j)} \in R_{\text{selected}}^{(i,j)}$ and $\chi_{p2}^{(i,j)} \in R_{\text{selected}}^{(i,j)}$, respectively; $g_{l\min}$ and $g_{l\max}$ are the minimal and maximal values of the lth elements, respectively; $\eta \in [0;1]$ is the coefficient that can be obtained randomly ($\eta = \text{rand}[0;1]$) or calculated depending on the values of the objective function $G_k^{(i,j)}$ of the parents $\chi_{p1}^{(i,j)}$ and $\chi_{p2}^{(i,j)}$ in order to increase the influence of the solution with the best (smallest) value $G_k^{(i,j)}$ as shown below:

$$\eta = \begin{cases} \text{rand}[0{,}5;1], & G_{p1}^{(i,j)} < G_{p2}^{(i,j)}; \\ 1 - \text{rand}[0;0{,}5], & G_{p1}^{(i,j)} \geq G_{p2}^{(i,j)}. \end{cases} \tag{28}$$

It is reasonable to apply the latter expression when new solutions are formed using formula (24), while the random generation of the coefficient $\eta = \text{rand}[0;1]$ makes sense when formulas (25)–(27) are used. Using formulas (24)–(28) in the proposed crossover operator allows one to generate new solutions $\chi_k^{(i',j)}$ given the peculiarities of the adjustable parameters g_{lk} (range of their values) and the fitness of the parent solutions $\chi_{p1}^{(i,j)} \in R_{\text{selected}}^{(i,j)}$ and $\chi_{p2}^{(i,j)} \in R_{\text{selected}}^{(i,j)}$.

As noted above, the mutation operator brings $N_{\text{mutation}} = \gamma N_\chi$ solutions to the new set $R^{(i',j)}$. We perform mutations over the selected solutions $\chi_{\text{mutated}}^{(i,j)} \in R_{\text{selected}}^{(i,j)}$ in order to vary the values of some elements $g_{lk,\text{mutated}}^{(i)} \in \chi_{\text{mutated}}^{(i,j)}$ within some range. The number $N_{g,\text{mutated}}$ of mutating parameters $g_{lk,\text{mutated}}^{(i)}$ is determined as follows: $N_{g,\text{mutated}} = \vartheta N_g$, ϑ is the portion of parameters $g_{lk}^{(i)}$ that can mutate in the selected solutions $\chi_{\text{mutated}}^{(i,j)} \in R_{\text{selected}}^{(i,j)}$. We propose to setting the default portion ϑ of mutating parameters $g_{lk,\text{mutated}}^{(i)}$ at a level of 10% so it will be possible to vary the values of every tenth adjustable parameter. We propose the following expression for varying the values of $g_{lk}^{(i)}$:

$$g_{lk}^{(i')} = \begin{cases} \text{rand}[g_{l\min}; g_{l\max}], & G_{\text{mutated}}^{(i,j)} \geq G_{\text{mean}}^{(i,j)}; \\ \text{rand}[g_{l\min} - \Delta g_l; g_{l\max}], & G_{\text{mutated}}^{(i,j)} < G_{\text{mean}}^{(i,j)}, \ g_l \to c_{mn}; \\ \text{rand}[g_{l\min}; g_{l\max} + \Delta g_l], & G_{\text{mutated}}^{(i,j)} < G_{\text{mean}}^{(i,j)}, \ g_l \to d_{mn}; \\ \text{rand}[0; g_{lk}^{(i)}], & G_{\text{mutated}}^{(i,j)} < G_{\text{mean}}^{(i,j)}, \ g_l \to w_m^{(4,r)}, \end{cases} \tag{29}$$

where $\Delta g_l = g_{l\max} - g_{l\min}$ is the width of the range of change of the lth elements g_l; $G_{\text{mean}}^{(i,j)}$ is the mean value of the objective function $G_k^{(i,j)}$ in the set $R^{(i,j)}$; and the designations $g_l \to c_{mn}$, $g_l \to d_{mn}$, and $g_l \to w_m^{(4,r)}$ reflect the type of the adjustable parameter in the element g_l, i.e., c_{mn}, d_{mn}, and $w_m^{(4,r)}$, respectively.

Expression (29) allows one to generate the values of mutating parameters $g_{lk,\text{mutated}}^{(i)}$ as random numbers from the respective ranges $[g_{l\min}; g_{l\max}]$ if the value of the objective function $G_{\text{mutated}}^{(i,j)}$ does not exceed the mean value of $G_k^{(i,j)}$ in the set $R^{(i,j)}$. Otherwise, in addition to the peculiarities of the location (the range of values) of the adjustable parameters g_{lk}, we take into account their type (the parameters c_{mn}, d_{mn} of the membership functions of the first-layer neurons, and the weight coefficients $w_m^{(4,r)}$ of the fourth-layer neuroelements). It is allowed to go outside the corresponding ranges so that one can expand the search domain and go outside the of possible ranges of local extrema. Furthermore, it is ensured that some values of the weight coefficients of the fourth layer $w_m^{(4,r)} = \text{rand}[0; g_{lk}^{(i)}]$ will decrease so that these weights can be then zeroed, which simplifies the structure of the fuzzy neural model to be synthesized.

After the new sets of solutions $R^{(i',j)} = \left\{\chi_1^{(i',j)}, \chi_2^{(i',j)}, \ldots, \chi_{|R^{(i,j)}|}^{(i',j)}\right\}$ are generated, the transformations of type (19) $\chi_k^{(i',j)} \to NFN_k^{(i',j)}$ are performed and each solution $\chi_k^{(i',j)}$ is estimated $G_k^{(i',j)} = G(\chi_k^{(i',j)}) = G(NFN_k^{(i',j)})$. Then, the stopping criteria are checked, including whether the maximum number of iterations N_{it} is achieved on each jth process, whether the acceptable value G_{ac} of the objective function $\min(G_k^{(i')}) \leq G_{\text{ac}}$ is obtained, etc. If these criteria are not met, the set $R^{(i'',j)}$ is formed.

To introduce some variety, migrations of the solutions $\chi_k^{(i,j)}$ into the neighboring populations $R^{(i,j')}$, $j' \neq j$, are performed in the population $R^{(i,j)}$ with some periodicity L_{migr}. Each jth process broadcasts $N_{\text{elite}} = \alpha N_\chi$ most fitted solutions $\chi_{k,\text{elite}}^{(i,j)}$ with calculated values of their objective functions $G_{k,\text{elite}}^{(i,j)} = G\left(\chi_{k,\text{elite}}^{(i',j)}\right)$ and receives similar solutions $\chi_{k,\text{elite}}^{(i,j')}$ and $G_{k,\text{elite}}^{(i,j')}$ from other processes $j' \neq j$. Thus, each jth process forms the set $R_{\text{migrated}}^{(i,j)}$ of migrated solutions $\chi_{k,\text{migrated}}^{(i,j)} \in R_{\text{migrated}}^{(i,j)}$.

We propose bringing $N_{\text{elite}} = \alpha N_\chi$ solutions from $R_{\text{migrated}}^{(i,j)}$ to the subpopulation $R^{(i',j)}$. We propose choosing the solutions $\chi_{k,\text{migrated}}^{(i,j)} \in R_{\text{migrated}}^{(i,j)}$ to be added to $R^{(i',j)}$ proportionally to the values of the objective functions $G_{k,\text{migrated}}^{(i,j)} = G\left(\chi_{k,\text{migrated}}^{(i',j)}\right)$ as follows: if the condition $G_{k,\text{migrated,norm}}^{(i,j)} \leq \text{rand}[0;1]$ is met, the solution $\chi_{k,\text{migrated}}^{(i,j)} \in R_{\text{migrated}}^{(i,j)}$ is added to the subset $R^{(i',j)}$, where $G_{k,\text{migrated,norm}}^{(i,j)}$ is the normalized value of the objective function $G_{k,\text{migrated}}^{(i,j)}$ of the solution $\chi_{k,\text{migrated}}^{(i',j)}$. To ensure that the condition $\alpha + \beta + \gamma = 1$ is met, which allows one to leave the total number of solutions N_χ / N_{pr} on each jth process (in each subset $R^{(i,j)}$) unchanged is met, we need to reduce the value of the sum of coefficients β and γ by the variable α on random search iterations that transfer some solutions to other processes as follows: $(\beta + \gamma)_{\text{migrated}} = (\beta + \gamma) - \alpha$, which reduces the number of children generated by crossover and mutation.

After N_{it} iterations of the search $RS(R^{(i,j)})$ are performed in each of the N_{pr} subsets $R^{(i,j)}$, they are combined into a single population $R^{(i')} = R^{(i',1)} \cup R^{(i',2)} \cup \ldots \cup R^{(i',N_{\text{pr}})}$. To do this, the sets of values

$$\left\langle R^{(i',j)}, G^{(i',j)} \right\rangle, \quad R^{(i',j)} = \left\{ \chi_1^{(i',j)}, \chi_2^{(i',j)}, \ldots, \chi_{|R^{(i,j)}|}^{(i',j)} \right\}, \quad G^{(i',j)} = \left\{ G_1^{(i',j)}, G_2^{(i',j)}, \ldots, G_{|R^{(i,j)}|}^{(i',j)} \right\}, \quad j = 1, 2, \ldots, N_{\text{pr}}$$

are sent to the main process. Then, the random search $RS(R^{(i')})$ is performed over the combined population $R^{(i')}$ and the values of the objective functions are calculated on various processes of the parallel system. This search allows one to study the search space without decomposing it into some parts and find new extremal domains that contain suboptimal solutions. We propose using the selection, crossover, and mutation operators given above to form new solutions. To reduce the time of probabilistic optimization, we propose determining the values of the objective functions $G_k^{(i')} = G(\chi_k^{(i')}) = G(NFN_k^{(i')})$, since the process is computationally complex, on different nodes of the parallel system.

After the random search $RS(R^{(i')})$ is performed and the stopping criteria are not found, the set $R^{(i')}$ is decomposed into subsets again as follows:

$$R^{(i')} \to \left\{ R^{(i',1)}, R^{(i',2)}, \ldots, R^{(i',N_{\text{pr}})} \right\},$$

after which a search $RS(R^{(i',j)})$ for optimal solutions is carried out in each of them. The processes $RS(R^{(i',j)})$, $j = 1, 2, \ldots, N_{\text{pr}}$ and $RS(R^{(i')})$ are performed alternately until the stopping criteria are met for the search.

Thus, the proposed method for the parametric identification of fuzzy neural models is based on the probabilistic approach when searching for values of adjustable parameters and implies the distribution of the most resource-intensive stages among the nodes of the parallel computing system, which allows us to reduce the time of adjusting the parameters (values of the weight coefficients and parameters of the membership functions of neuroelements) of the neural models to be synthesized.

4. CONCLUSIONS

In this work, we consider the topical problem of automating the process of training fuzzy neural models using the given sets of observations.

The scientific novelty of the work is that we proposed a method for the parametric identification of fuzzy neural networks based on parallel random search that uses probabilistic optimization to adjust the parameters of the models to be synthesized (parameters of the membership functions and weight coefficients of neuroelements) and forms the initial set of solutions based on information on the training sample (the significance of the terms of features is taken into account using the density of the location of instances of the training set in the respective term and its degree of influence on the value of the output parameter), which allows the initial search points to be brought closer to the optimal values and accelerates the optimization process.

To study the domains of local optima in more detail, the proposed method decomposes the current set of solutions into subsets; then, optima are searched for in each subset based on the corresponding processes of the parallel system. The respective subsets are formed so that each set represents a group of solutions densely located in the space of adjustable elements, which allows one to take into account the information on the spatial location of solutions and study domains of location of extremal solutions in more detail via random search.

The developed operators of forming the new set of solutions take into account the peculiarities of the location (the range of values) of adjustable parameters, their type (parameters of membership functions of neurons, weight coefficients of neuroelements), and the fitness of parent solutions, as well as allow one to go outside of the respective ranges so that we can expand the search domain and go outside of the possible domains of local extrema.

We highlight the following directions for future research:

—a theoretical analysis of computational complexity and an estimation of the efficiency of the proposed parallel method of training fuzzy neural networks,

—experimental research on a cluster and vector processor when solving practical problems that require constructing diagnostic neural network models.

REFERENCES

1. Ding, S.X., *Model-Based Fault Diagnosis Techniques: Design Schemes, Algorithms, and Tools.* Berlin: Springer-Verlag, 2008.
2. *ASM Handbook. Vol. 17: Nondestructive Evaluation and Quality Control.* Cleveland: ASM International, 1997.
3. Blanke, M., Kinnaert, M., Lunze, J., and Staroswiecki, M., *Diagnosis and Fault-Tolerant Control.* Berlin: Springer-Verlag, 2006.
4. Vachtsevanos, G., Lewis, F., Roemer, M., Hess, A., and Wu, B., *Intelligent Fault Diagnosis and Prognosis for Engineering Systems.* New Jersey: Wiley, 2006.
5. Jang, J.R., Sun, C.-T., and Mizutani, E., *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence.* Upple Saddle River: Prentice-Hall, 1997.
6. Nauck, D., Klawonn, F., and Kruse, R., *Foundations of Neuro-Fuzzy Systems.* Chichester: Wiley, 1997.
7. Rutkowski, L., *Flexible Neuro-Fuzzy Systems: Structures, Learning and Performance Evaluation.* Boston: Kluwer, 2004.
8. Yu, X. and Gen, M., *Introduction to Evolutionary Algorithms (Decision Engineering).* London: Springer-Verlag, 2010.
9. *Encyclopedia of Machine Learning,* Sammut, C. and Webb, G.I., Eds., New York: Springer-Verlag, 2011.
10. Baragona, R., Battaglia, F., and Poli, I., *Evolutionary Statistical Procedures: An Evolutionary Computation Approach to Statistical Procedures Designs and Applications.* Berlin: Springer-Verlag, 2011.
11. Smith, S. and Cagnoni, S., *Genetic and Evolutionary Computation: Medical Applications.* Chicehster: Wiley, 2011.
12. Jang, J.R., ANFIS: adaptive-network-based fuzzy inference system, *IEEE Trans. on Systems Cybernetics.* 1993, vol. 23, pp. 665–685.
13. Bishop, C.M., *Pattern Recognition and Machine Learning.* New York: Springer-Verlag, 2006.
14. Zadeh, L.A., Fuzzy logic, neural networks, and soft computing, *Commun. ACM.* 1994, vol. 37, no. 3, pp. 77–84.
15. Kandel, A. and Langholz, G., *Fuzzy Control Systems.* Cleveland: CRC, 1993.
16. *Encyclopedia of Artificial Intelligence,* Dopico, J.D. and de la Calle, A.P., Eds. New York: Information Science Reference, 2009.

Translated by M. Talacheva